УДК681.326:519. 713

СИНТЕЗ КУБИТНЫХ ПОКРЫТИЙ ДЛЯ ЦИФРОВЫХ СИСТЕМ

БОЯДЖЯН А.Г., КОТЛЯРОВ А.С.

Описываются квантовые компьютеры, которые могут решать проблемы, трудно разрешимые даже для самых мощных современных суперкомпьютеров. Они имеют потенциал для решения проблем в области материаловедения, химии и математики, которые далеко не подходят для суперкомпьютеров. Их мощность обусловлена использованием квантовых битов, которые могут одновременно существовать как в 0, так и в квантовом состоянии суперпозиции. Синтез кубических покрытий для цифровых систем является основополагающей парадигмой для квантового компьютинга [1,2].

Ключевые слова: кубит; квантовый компьютинг; синтез; цифровая система.

Key words: qubit; quantum computing; synthesis; digital system.

1. Введение

Понятие адресного выполнения логических операций, реализованных на элементах памяти LUT в программируемых логических устройствах (PLD), дает потенциальную возможность создавать на кристалле только адресное пространство, максимально технологичное для встроенного восстановления работоспособности всех компонентов, участвующих в формировании функциональности [5, 6].

Тенденция к увеличению памяти влечет возможность встроенного восстановления работоспособности отказавших ячеек за счет выделенных дополнительных ресурсов для их ремонта (sparelogiccells). Проблема автономного устранения дефектов (самовосстановления работоспособности) логических элементов связана с отсутствием у них адресов. Но решить ее можно, если связи между элементами логики сделать гибкими с помощью программы описания структуры, помещенной в память, которая соединит логические компоненты в схему. Кроме структуры взаимодействия элементов, память должна содержать порядок их обработки. В случае возникновения дефекта в одном из адресуемых логических элементов система встроенного тестирования восстановит его работоспособность путем переадресации на заведомо исправный аналог из ремонтного запаса. Таким образом, решается проблема повышения качества и надежности цифровых систем на кристаллах путем создания инфраструктуры встроенного тестирования, диагностирования, оптимизации и восстановления работоспособности за счет аппаратной избыточности и уменьшения быстродействия выполнения функциональных операций [3,7].

Другими словами, любой вычислительный процесс сводится к адресным операциям считывания-записи на памяти. Практическая значимость данной парадигмы заключается в исключении аппаратной логики из компьютинга, что дает возможность:

1) Сделать регулярной архитектуру компьютера на основе использования только матриц памяти.

2) Выполнять все арифметические, логические и специальные операции на матричной структуре памяти без обращения к внешнему модулю АЛУ, который значительно уменьшает быстродействие.

3) Технологично решать все проблемы надежности компьютерных систем на основе встроенного online-тестирования и ремонта отказавших модулей памяти путем их переадресации на исправные из ремонтного запаса.

4) Иметь неограниченные возможности в пределах одного кристалла для параллельного выполнения логических операций любой регистровой размерности.

5) Существенно уменьшать время проектирования специализированных процессоров за счет исключения сложных и нерегулярных блоков АЛУ с шинами обмена информацией.

6) Перепрограммировать в режиме online логику элементов памяти для выполнения других операций [5, 6].

Далее рассмотрим синтез кубических покрытий для цифровых устройств. На примерах будет показано, как можно проводить процесс синтеза схемы на памяти. Предлагается существенное уменьшение времени при синтезе цифровых схем.

Цель – показать технологичность и производительность методов и алгоритмов, использующих кубитные покрытия функциональных примитивов.

Задачи: 1) Синтез кубитных моделей цифровых схем.2) Минимизациякубитных покрытий. 3) Верификация метода и разработка алгоритма синтеза моделей для их имплементации в программный сервис.

2. Кубитная форма описания примитивов

Кубит (n-кубит) есть векторная форма унитарного кодирования универсума из n примитивов для задания булеана состояний 2^2^n с помощью 2^n двоичных переменных. Если n=2, то 2-кубит задает 16 состояний с помощью четырех переменных, при n=1, кубит задает четыре состояния на универсуме из двух примитивов (10) и (01) с помощью двух двоичных переменных (00,01,10,11). При этом допускается суперпозиция в векторе 2^n состояний, обозначенных примитивами. Синонимом кубита при задании двоичного вектора логической функции является Q-покрытие (Q-вектор), как унифицированная векторная форма суперпозиционного задания вы-

ходных состояний, соответствующих адресным кодам входных переменных функционального элемента. Формат структурного кубитного компонента цифровой схемы Q* = (X, Q, Y) включает интерфейс (входные и выходную переменные), а также кубит-вектор Q, задающий функцию Y = Q(X), размерность которого определяется степенной функцией от числа входных линий k = 2^n [6]. Новизна кубитной формы заключается в замене неупорядоченных по строкам таблиц истинности функциональных элементов векторами упорядоченных состояний выходов. Например, если функциональный примитив имеет двоичную таблицу, то ему можно поставить в соответствие кубит или Q- покрытие: Q = (1110) [4-6]:

$$C = \begin{array}{ccc} X_1 & X_2 & Y \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{array} \rightarrow Q = (1110).$$

Таким образом, идея, положенная в основу исследований, заключается в замене множества вход-выходных соответствий таблицы истинности кубитным вектором адресуемых выходных состояний (рис. 1) [6].

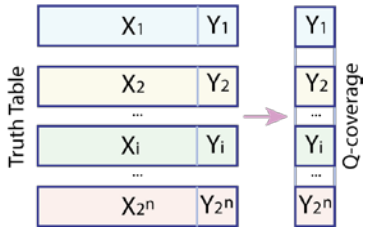

Рис. 1. Переход от таблицы истинности к Q-покрытию

Примитивизм и компактность кубитной векторной формы (см. рис.1) диктует применение только простых параллельных регистровых операций над его содержимым (not, shift, or, and, xor) для решения всех задач синтеза и анализа цифровых изделий.

3. Синтез кубитных моделей цифровых схем и минимизация кубитных покрытий

На рис. 2 представлена функция, аналитическая запись которой имеет вид:

$$f(x)= \sim X_2 \sim X_3 v X_1 X_2 X_3$$

1) Записываем связь входных линий и выходного значения, получая таблицу истинности каждого элемента (табл.1).

2) Для каждого логического элемента составляем кубит или вектор Q-покрытия (табл.2).

3) Используя третий элемент логического или, формируем выходной вектор, используя кубиты заданных логических элементов 1000 и 00000001, которые являются входом для элемента 0111.

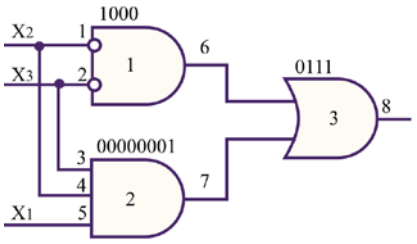

Рис. 2. Схема 1

4) Выполняем декартово произведение двух кубитов 1000 и 00000001.

Таблица 1. Таблица истинности элементов

3	2	6
0	0	1
0	1	0
1	0	0
1	1	0

3	4	5	7
0	0	0	0
0	0	1	0
.	.	.	.
1	1	1	1

6	7	8
0	0	0
0	1	1
1	0	1
1	1	1

Таблица 2. Q-покрытия элементов

6		
0	0	1
0	1	0
1	0	0
1	1	0

7			
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

8		
0	0	0
0	1	1
1	0	1
1	1	1

5) В итоге получаем результирующий вектор вида: 11111111 00000001 00000001 00000001. Для наглядности данный вектор разбит побайтово. Результат приведен в формуле:

1000v00000001										
	=	1111	1111	0000	0001	0000	0001	0000	0001	
	a 1	0000	0000	0000	0000	1111	1111	1111	1111	
	д 2	0000	0000	1111	1111	0000	0000	1111	1111	(1)
	р 3	0000	1111	0000	1111	0000	1111	0000	1111	
	е 4	0011	0011	0011	0011	0011	0011	0011	0011	
	c	0101	0101	0101	0101	0101	0101	0101	0101	

6) Поскольку вектор получился 32-разрядным, то необходимо закодировать в двоичном коде 32 адреса от 00000… до 11111. Для этого записываем по вертикали адреса под каждым битом получившегося вектора.

7) Анализ схемы показывает, что линия 2 и 3 относятся к одному входу, как и линии 1 и 4. Так как одновременно на линиях не могут быть разные сигналы, то необходимо объединить линии в адресах 1,4 и 2,3.

8) В заданных линиях нужно оценить, где в них отличаются значения, и выделить их. После того как все столбцы с различающимися значениями будут выделены, смотрим на те столбцы, которые не были затронуты. Иначе говоря, используя операцию xor, необходимо обнаружить одинаковые значения и исключить различающиеся значения. После проведения операции xor получаем результирующий вектор: 11000001.

9) Для проверки результата необходимо посмотреть какой вектор получился на выходе линии 8. Стоит обратить внимание, что последовательность размещения входных сигналов имеет значение и она должна быть, согласно схеме, X_2, X_3, X_1.

10) Проверяем результат, построив таблицу истинности (табл.3). Результат получается аналогичным 11000001, что показывает работоспособность данного метода.

Таблица 3. Таблица истинности элемента 0111

X_2	X_3	X_1	6	7	8
0	0	0	1	0	1
0	0	1	1	0	1
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	0	1	1

Смоделируем исправное поведение на основе входного сигнала 100, используя структуры данных, представленных в табл. 4.

Таблица 4. Модель для анализа цифровой системы на основе структур данных

L	1	2	3	4	5	6	7	8
M	0	0	0	0	1	1	0	1
X	.	.	.	.	.	12	345	67
Q	.	.	.	.	.	1	0	0
	.	.	.	.	.	0	0	1
	.	.	.	.	.	0	0	1
	.	.	.	.	.	0	0	1
	.	.	.	.	.	.	0	.
	.	.	.	.	.	.	0	.
	.	.	.	.	.	.	0	.
	.	.	.	.	.	.	1	.

Здесь представлены соответственно: L – вектор идентификаторов эквипотенциальных линий схемы цифровой системы, который ввиду своей тривиальности может быть исключен из модели, но при этом необходимо иметь число входных переменных устройства и общее количество линий; M – вектор моделирования состояний всех линий схемы; X – упорядоченная совокупность векторов входных переменных каждого примитива схемы, привязанных к номерам выходов; Q – совокупность q-покрытий примитивов, строго привязанных к номерам выходов и входным переменным примитивов.

Фактически метод позволяет реализовать синтез кубических покрытий для цифровых устройств. Адреса, по которым проводится операция xor,могут быть перенесены на память, поэтому скорость работы схемы увеличивается во много раз, так как процесс занимает не более нескольких тактов. На простых схемах разница между быстродействием может быть несущественной, но, взяв более сложную схему, можно увидеть разницу. К сожалению, более сложные схемы ручным методом довольно сложно описать, так как на примере схемы Шнейдера результирующий вектор будет 256-разрядным. Для того чтобы доказать работоспособность метода, на второй схеме (рис. 3) представлена другая, более сложная схема, аналитическая запись которой имеет следующий вид:

$$f= X_1\ X_2 \vee (X_3 \text{ xor } X_4) \vee (X_1 \vee X_4).$$

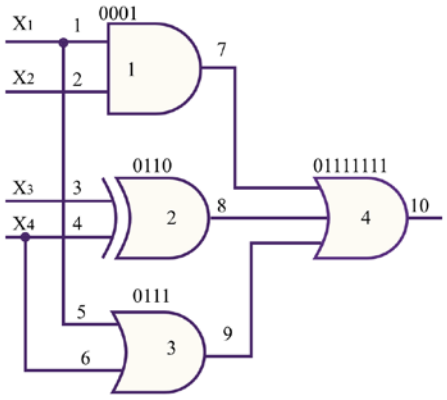

Рис. 3. Схема 2

1) Построение таблицы истинности (табл. 5).

Таблица 5. Таблица истинности элементов

1	2	7
0	0	0
0	1	0
1	0	0
1	1	1

3	4	8
0	0	0
0	1	1
1	0	1
1	1	0

1	4	9
0	0	0
0	1	1
1	0	1
1	1	1

2) Построение Q-покрытий (табл.6).

Таблица 6. Q-покрытия элементов

7			8			9		
0	0	0	0	0	0	0	0	0
0	1	0	0	1	1	0	1	1
1	0	0	1	0	1	1	0	1
1	1	1	1	1	0	1	1	1

3) Используя третий элемент логического или, формируем выходной вектор, используя кубиты заданных логических элементов 0001, 0110, 0111, которые являются входом для элемента 01111111.

4) Выполняем декартово произведение трёх векторов. В итоге получаем вектор вида: 01111111 11110111 01111111 11110111 01111111 11110111 11111111 11111111. Для наглядности данный вектор разбит побайтово. Результат приведен в формуле (2).

5) Вектор получился 64-разрядным, необходимо закодировать в двоичном коде 64 адреса от 000000… до 111111. Для этого записываем по вертикали адреса под каждым битом получившегося вектора.

6) Проанализировав схему, видим, что линия 1 и 5 относятся к одному входу, как и линии 4 и 6. Так как одновременно на линиях не могут быть разные сигналы, то необходимо объединить линии в адресах 1,5 и 4,6.

7) В заданных линиях нужно оценить, где в них отличаются значения, и выделить их.

8) Используя операцию xor, необходимо обнаружить одинаковые значения и исключить различающиеся. После проведения операции xor получается результирующий вектор: 0111 0111 1111 1111.

9) Построив таблицу истинности, видим, что результат получаем аналогичным 0111 0111 1111 1111, что является проверкой данного метода (табл.7).

В табл. 8 показано моделирование исправного поведения на основе входного сигнала 0011.

4. Алгоритм синтеза моделей

Для имплементации в программный сервис алгоритм будет иметь следующий вид (рис.4):

1. Получение таблицы истинности для каждого логического элемента, согласно входам и выходу, относящегося к элементу.
2. Составление кубита или вектора Q- покрытия для каждого логического элемента.
3. Формирование выходного вектора с помощью декартова произведения логических элементов.
4. Кодирование адресов в зависимости от разрядности результирующего вектора.
5. Изучение входов схемы и объединение линий, относящихся к одному и тому же элементу.
6. Обнаружение одинаковых значений и исключение отличающихся с использованием операции xor.
7. Проверка результата с помощью построения таблицы истинности на выход.

Таблица 7. Таблица истинности элемента 01111111

X_1	X_2	X_3	X_4	7	8	9	10
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	0	1	0	1
0	0	1	1	0	0	1	1
0	1	0	0	0	0	0	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	0	1	1
1	0	0	0	0	0	1	1
1	0	0	1	0	1	1	1
1	0	1	0	0	1	1	1
1	0	1	1	0	0	1	1
1	1	0	0	1	0	1	1
1	1	0	1	1	1	1	1
1	1	1	0	1	1	1	1
1	1	1	1	1	0	1	1

Таблица 8. Модель для анализа цифровой системы на основе использования кубитных структур данных

L	1	2	3	4	5	6	7	8	9	10
M	0	0	1	1	0	1	0	0	1	1
X	.	.	.	.	.	.	12	34	56	789
Q	.	.	.	.	.	.	0	0	0	0
	.	.	.	.	.	.	0	1	1	1
	.	.	.	.	.	.	0	1	1	1
	.	.	.	.	.	.	1	0	1	1
	.	.	.	.	.	.	.	.	.	1
	.	.	.	.	.	.	.	.	.	1
	.	.	.	.	.	.	.	.	.	1
	.	.	.	.	.	.	.	.	.	1

0001 v 0110 v 0111 =

	01111111 11110111 01111111 11110111 01111111 11110111 11111111 11111111	
а	00000000 00000000 00000000 00000000 11111111 11111111 11111111 11111111	
д	00000000 00000000 11111111 11111111 00000000 00000000 11111111 11111111	
р	00000000 11111111 00000000 11111111 00000000 11111111 00000000 11111111	(2)
е	00001111 00001111 00001111 00001111 00001111 00001111 00001111 00001111	
с	00110011 00110011 00110011 00110011 00110011 00110011 00110011 00110011	
	01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101	

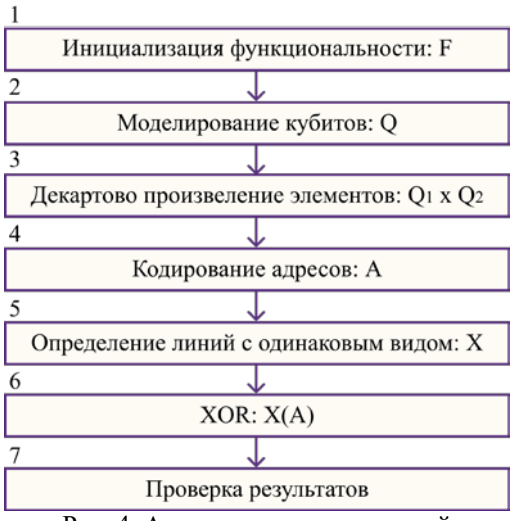

Рис. 4. Алгоритм синтеза моделей

5. Заключение

В результате исследования синтеза кубических покрытий для цифровых устройств был разработан алгоритм с существенно уменьшенным временем синтеза путем выполнения операций на матричной структуре памяти, без обращения к внешнему модулю АЛУ, которое значительно уменьшает быстродействие. Актуальность работы заключается в том, что вычисления строятся на векторе и выполняются за такт. Показано насколько технологичен и производителен данный алгоритм. Учитывая нехватку вычислительной мощности при обработке данных, к примеру,BigData, данный алгоритм поможет ускорить процессы обработки информации.

Литература: **1.** *Almudever C. G., Lao TL., Fu X., Khammassi N., Ashraf I., Iorga D., Varsamopoulos S., Eichler C., Wallraff A., Geck L., Kruth A., Knoch J., Bluhm H., Bertels K.* The engineering challenges in quantum computing // 2017 Design, Automation & Test in Europe Conference & Exhibition (DATE). Lausanne, Switzerland. 27-31 March 2017. DOI: 10.23919/DATE.2017.7927104. **2.** *Lieven Vandersypen, Antoni van Leeuwenhoek.* 1.4 Quantum computing - the next challenge in circuit and system design // *Solid-State Circuits Conference (ISSCC), 2017 IEEE International.* San Francisco, CA, USA. 5-9 February. 2017. DOI: 10.1109/ISSCC.2017.7870244.**3.** *Jasmeet Singh, Mohit Singh.* Evolution in Quantum Computing // *System Modeling & Advancement in Research Trends (SMART), International Conference*. Moradabad, India. 25-27 November 2016. DOI: 10.1109/SYSMART.2016.7894533.**4.** *Travis S. Humble, Keith A. Britt.* Software systems for high-performance quantum computing // *High Performance Extreme Computing Conference (HPEC)*, 2016 IEEE. Waltham, MA, USA. 13-15 September 2016. DOI: 10.1109/HPEC.2016.7761628 **5.** *Hahanov V., Litvinova E., Chumachenko S.* Cyber Physical Computing for IoT Driven Sevices. *Springer International Publisher*. Switzerland. 2017. 150 p. **6.** *Vladimir Hahanov, Svetlana Chumachenko, Eugenia Litvinova, Mykhailo Liubarskyi.* Qubit Description of the Functions and Structures for Computing // *Proc. of IEEE East-West Design and Test Symposium*. Yerevan. 14-17 October 2016. P.88-93.
7. *http://www.gartner.com/newsroom/id/3784363.*

Transliterated bibliography:

1. *Almudever C. G., Lao TL., Fu X., Khammassi N., Ashraf I., Iorga D., Varsamopoulos S., Eichler C., Wallraff A., Geck L., Kruth A., Knoch J., Bluhm H., Bertels K.* The engineering challenges in quantum computing. // 2017 Design, Automation & Test in Europe Conference & Exhibition (DATE). Lausanne, Switzerland. 27-31 March 2017. DOI: 10.23919/DATE.2017.7927104.
2. *LievenVandersypen, Antoni van Leeuwenhoek.* 1.4 Quantum computing - the next challenge in circuit and system design // Solid-State Circuits Conference (ISSCC), 2017 IEEE International.San Francisco, CA, USA. 5-9 February. 2017. DOI:10.1109/ISSCC.2017.7870244.
3. *Jasmeet Singh, Mohit Singh.* Evolution in Quantum Computing // System Modeling & Advancement in Research Trends (SMART), International Conference. Moradabad, India. 25-27 November 2016. DOI: 10.1109/SYSMART.2016.7894533.
4. *Travis S. Humble, Keith A. Britt.* Software systems for high-performance quantum computing // High Performance Extreme Computing Conference (HPEC), 2016 IEEE. Waltham, MA, USA. 13-15 September 2016. DOI: 10.1109/HPEC.2016.7761628
5. *Hahanov V., Litvinova E., Chumachenko S.* Cyber Physical Computing for IoT Driven Sevices. Springer International Publisher. Switzerland. 2017. 250 p.
6. *VladimirHahanov, SvetlanaChumachenko, EugeniaLitvinova, MykhailoLiubarskyi.* Qubit Description of the Functions and Structures for Computing // *East-West Design & Test Symposium (EWDTS), 2016 IEEE*. Yerevan, Armenia. 14-17 October 2016. P.88-93.
7.*http://www.gartner.com/newsroom/id/3784363.*

Поступила в редколлегию 17.05.2017
Рецензент: д-р техн. наук, проф. Хаханова И.В.

Бояджян Армен Гамлетович, студент 6 курса ХНУРЭ. Научные интересы: Internetofthings, квантовые вычисления. Адрес: Украина, 61166, Харьков, пр. Науки, 14, e-mail: z@fastlive.eu.
Котляров Антон Сергеевич, студент 6 курса ХНУРЭ. Научные интересы: Internetofthings, квантовые вычисления.Адрес: Украина, 61166, Харьков, пр. Науки, 14, e-mail: kotlyarovanton2@gmail.com.
Boiadzhian Armen, 6th year student of Kharkiv National University of Radioelectronics. Research interests: Internet of things, quantum computing. Address: Ukraine, 61009, Kharkiv, Nauki Ave., 14, e-mail: z@fastlive.eu.
Kotliarov Anton, 6th year student of Kharkiv National University of Radioelectronics. Research interests: Internet of things, quantum computing. Address: Ukraine, 61009, Kharkiv, Nauki Ave., 14.